Limited-Memory Techniques for Sensor Placement in Water Distribution Networks

William E. Hart, Jonathan W. Berry, Erik Boman, Cynthia A. Phillips,
Lee Ann Riesen, and Jean-Paul Watson

Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185 USA
{wehart, jberry, egboman, lafisk, caphill, jwatson}@sandia.gov

Abstract. The practical utility of optimization technologies is often impacted by factors that reflect how these tools are used in practice, including whether various real-world constraints can be adequately modeled, the sophistication of the analysts applying the optimizer, and related environmental factors (e.g. whether a company is willing to trust predictions from computational models). Other features are less appreciated, but of equal importance in terms of dictating the successful use of optimization. These include the scale of problem instances, which in practice drives the development of approximate solution techniques, and constraints imposed by the target computing platforms. End-users often lack state-of-the-art computers, and thus runtime and memory limitations are often a significant, limiting factor in algorithm design. When coupled with large problem scale, the result is a significant technological challenge. We describe our experience developing and deploying both exact and heuristic algorithms for placing sensors in water distribution networks to mitigate against damage due intentional or accidental introduction of contaminants. The target computing platforms for this application have motivated limited-memory techniques that can optimize large-scale sensor placement problems.

1 Introduction

Real-world optimization problems are often complicated by factors that make them more challenging than problem formulations that are studied by academic researchers. For example, industrial scheduling problems differ significantly from academic scheduling models like job-shop and resource-constrained scheduling problems because of large numbers of company-specific constraints. Most researchers acknowledge the impact of such side constraints, but what is far less appreciated is the degree to which factors like problem size and target computational platform impact the practical solution of real-world optimization problems. In this paper we present a real-world case study that highlights how memory limitations impact a deployed solution technology.

Our case study involves the protection of drinking water distribution systems, found in municipalities throughout the world. Public water distribution systems

are inherently vulnerable to accidental or intentional contamination because of their distributed geography. The use of on-line, real-time contaminant warning systems (CWSs) is a promising strategy for mitigating these risks. The general goal of a CWS is to identify a low-probability, high-impact contamination incident while allowing sufficient time for an appropriate response that mitigates adverse impacts. A CWS may complement conventional routine monitoring by quickly providing information on unusual threats to a water supply.

A key element of the design of an effective CWS is the strategic placement of sensors throughout the distribution network. We have recently demonstrated that a canonical sensor placement formulation is equivalent to the well-known p-median facility location problem. However, the p-median problems that arise in real-world sensor placement applications are much larger than typical facility location instances considered in the literature. For example, the largest p-median instances commonly investigated are limited to approximately 10,000 facilities and customers. In contrast, the p-median instances arising in real-world CWS design can involve as many as 50,000 facilities and hundreds of thousands of customers. The magnitude of these instances requires efficient computational techniques to deal with the increase in solution difficulty. Further, these large-scale instances have significant memory footprints that exceed the limits available in the computing platforms of most water utilities.

We illustrate how memory limitations have influenced the development of sensor placement algorithms in the TEVA-SPOT Toolkit [10] (SPOT). SPOT provides a sensor placement framework that facilitates research in sensor placement optimization and enables the practical application of sensor placement solvers to real-world CWS design applications. SPOT contains algorithms for solving the integer programming formulation exactly (e.g., via CPLEX), heuristically via GRASP, and heuristically via Lagrangian relaxation. The United States Environmental Protection Agency (USEPA) has funded the development of SPOT to support the analysis of US water distribution networks in the USEPA TEVA program [16]. SPOT's support of limited-memory sensor placement techniques has been crucial for the successful analysis of these large-scale networks.

This paper is organized as follows. We begin in Section 2 with a detailed description of the CWS problem and the corresponding integer programming formulation. We discuss limited-memory strategies for integer programming, GRASP and Lagrangian relaxation in Sections 3, 4, and 5 respectively. We conclude in Section 6 with a discussion of the implications of our results for real-world problem solving.

2 Background

Contamination warning systems (CWSs) have been proposed as a promising approach for detecting contamination incidents in drinking water distribution systems. The goal of a CWS is to detect contamination incidents early enough to allow for effective public health and/or water utility intervention to limit potential public health or economic impacts. There are many challenges to de-

tecting contaminants in drinking water systems: municipal distribution systems are large, consisting of hundreds or thousands of miles of pipe; flow patterns are driven by time-varying demands placed on the system by customers; and distribution systems are looped, resulting in mixing and dilution of contaminants. The drinking water community has proposed that CWSs be designed to maximize the number of contaminants that can be detected in drinking water distribution systems by combining online sensors with public health surveillance systems, physical security monitoring, customer complaint surveillance, and routine sampling programs [22].

For CWS design, the general goal of sensor placement optimization is to place a limited number of sensors in a water distribution network such that the impact to public health of contaminant injection is minimized. However, there is no specific formulation of the problem that is widely accepted by the water resources management community. There are a wide range of important design objectives for sensor placements (e.g., minimizing the cost of sensor installation and maintenance, the response time to a contamination incident, and the extent of contamination), and researchers have developed different formulations when studying these objectives. Further, researchers have developed a variety of technical approaches for solving sensor placement problems including mixed-integer programming (MIP) models [6, 5, 14, 13, 19, 23], combinatorial heuristics [11, 12, 17], and general-purpose metaheuristics (e.g., [17]).

A common feature of most sensor placement formulations is that they rely either directly or indirectly on contaminant transport simulation models. Simulation tools, like EPANET [21], perform extended-period simulation of the hydraulic and water quality behavior within pressurized pipe networks. These models can be used to evaluate the expected flow in water distribution systems, and they can model the transport of contaminants and related chemical interactions. Thus, a water utility can assess risks to their distribution network by considering simulations of an ensemble of contamination incidents, which reflect the impact of contamination at different locations, times of the day, etc.

A key limitation of early sensor placement formulations is that they incorporate contamination transport simulation results indirectly. Consequently, the optimized value of the final solution may not accurately approximate a risk assessment performed with contaminant transport simulations. We have proposed a mixed-integer programming (MIP) model that resolves this difficulty by directly integrating contaminant transport simulation results [7, 6]. The MIP objective exactly captures water utilities' current risk metrics. Furthermore, this model can minimize a variety of different design objectives simply by integrating different statistics from the simulation results. This model assumes that a potentially large number of contamination incidents can be simulated, but these simulations are preprocessing steps that can be done in advance of the optimization process. Thus, the time needed for simulation does not impact the time spent performing sensor placement.

Our MIP formulation for sensor placement is:

$$\text{(SP) minimize} \sum_{a \in \mathcal{A}} \alpha_a \sum_{i \in \mathcal{L}_a} d_{ai} x_{ai}$$

$$\text{where} \begin{cases} \sum_{i \in \mathcal{L}_a} x_{ai} = 1 & \forall a \in \mathcal{A} \\ x_{ai} \leq s_i & \forall a \in \mathcal{A}, i \in \mathcal{L}_a \\ \sum_{i \in L} s_i \leq p & \\ s_i \in \{0,1\} & \forall i \in L \\ 0 \leq x_{ai} \leq 1 & \forall a \in \mathcal{A}, i \in \mathcal{L}_a \end{cases}$$

This MIP minimizes the expected impact of a set of contamination incidents defined by $\mathcal{A}$. For each incident $a \in \mathcal{A}$, α_a is the weight of incident a, frequently a probability. The EPANET simulator reports contamination levels at a set of *locations*, denoted L, where a location refers to network junction. For each incident a, $\mathcal{L}_a \subseteq L$ is the set of locations that can be contaminated by a. Thus a sensor at a location $i \in \mathcal{L}_a$ can detect contamination from incident a at the time contamination first arrives at location i. Each incident is *witnessed* by the first sensor to see it. For each incident $a \in \mathcal{A}$ and location $i \in \mathcal{L}_a$, d_{ai} defines the impact of the contamination incident a if it is witnessed by location i. This impact measure assumes that as soon as a sensor witnesses contamination, then any further contamination impacts are mitigated (perhaps after a suitable delay that accounts for the response time of the water utility). The s_i variables indicate where sensors are placed in the network, subject to a budget p, and the x_{ia} variables indicate whether incident a is witnessed by a sensor at location i.

We may not be able to witness all contamination incidents with a given set of sensors. To account for this, L contains a *dummy* location. This dummy location is in all subsets $\mathcal{L}_a$. The impact for this location is the impact of the contamination incident after the entire contaminant transport simulation has finished, which corresponds to the impact that would occur without an online CWS.

Remarkably, SP is identical to the well-known p-median facility location problem [15]. In the p-median problem, p facilities (e.g., central warehouses) are to be located on m potential sites such that the sum of distances d_{ai} between each of n customers (e.g., retail outlets) and the nearest facility i is minimized. In comparing SP and p-median problems, we observe equivalence between (1) sensors and facilities, (2) contamination incidents and customers, and (3) contamination impacts and distances. While SP allows placement of *at most* p sensors, p-median formulations generally enforce placement of all p facilities; in practice, the distinction is irrelevant unless p approaches the number of possible locations.

The flexibility of this sensor placement formulation is illustrated by the TEVA-SPOT Toolkit (SPOT) [10], which integrates a variety of sensor placement solvers developed by Sandia National Laboratories and the Environmental Protection Agency, along with many academic collaborators [5, 7, 6]. SPOT includes general-purpose heuristic solvers, integer programming heuristics, exact solvers, and linear-programming bounding techniques. SPOT can place sensors to minimize a variety of design objectives, including population-based public health measures, time to detection, extent of pipe contamination, volume consumed, and number of failed detections.

The size of the SP formulation is largely a function of the the total number of impacts, D. This is the dominant term in the number of constraints, the number of variables, and the number of nonzeros in the constraint matrix. We can compute a lower bound on SP by relaxing the integrality constraints on the variables and solving the resulting *linear program* (LP). Solving the LP involves linear algebra. Dense methods would require space proportional to D^2. Solvers use sparse methods as much as possible. However, the space requirements are generally superlinear in D. Because integer-programming solvers use LPs for bounding subproblems, they require at least as much space as LP asymptotically.

Water distribution networks analyzed in the TEVA program have 1,000s to 10,000s of pipes and junctions. Due to memory limitations, contamination incidents are typically restricted to a small number of times during a day (e.g. morning, afternoon, evening and night incidents). The number of locations contaminated by an incident can be highly variable; although many incidents impact a small number of locations, some large networks have many incidents that contaminate a large fraction of the network. Many of the SP analyses performed in the TEVA program have had millions of impact values. Very large sensor placement problems considered in the TEVA program have had over 40,000 potential sensor placement locations, 20,000 contamination incidents and close to 30 million impact values. Furthermore, real-world analyses will ultimately require the consideration of many more contamination incidents, for example to model changes in weekday vs. weekend demands, as well as seasonal changes in demands.

3 Integer Programming

The SP MIP model provides a generic approach for performing sensor placement with a variety of design objectives. However, the size of this MIP formulation can quickly become prohibitively large, especially for 32-bit computers (yielding a maximum of 4GB of RAM in the case of Unix systems, and in practice 3GB of RAM in the case of Windows systems). As noted in the previous section, SP can require millions of impact values for large water distribution systems.

In Berry et al. [4, 7], we note that for any given contamination incident a, there are often many impacts d_{ai} that have the same value. If the contaminant reaches two junctions at approximately the same time, then the impact for these two junctions would have the same impact values. For example, this occurs frequently when we use a coarse reporting time-step for the water quality simulation.

This observation led to a revised formulation that treats sensor placement locations as equivalent if their corresponding contamination impacts are the same for a given contamination incident. Let $\mathcal{L}_{ai}$ be a maximal set of locations in $\mathcal{A}$ that all have the same impact for incident a. Considering any witness in $\mathcal{L}_{ai}$ equivalent reduces the set of effective witness "locations" to a new set $\hat{\mathcal{L}}_a$. The new MIP formulation is:

$$\text{(waSP) minimize} \sum_{a \in \mathcal{A}} \alpha_a \sum_{i \in \hat{\mathcal{L}}_a} d_{ai} x_{ai}$$

$$\text{where} \begin{cases} \sum_{i \in \hat{\mathcal{L}}_a} x_{ai} = 1 & \forall a \in \mathcal{A} \\ x_{ai} \leq \sum_{j \in \hat{\mathcal{L}}_{ai}} s_j & \forall a \in \mathcal{A}, i \in \hat{\mathcal{L}}_a \\ \sum_{i \in L} s_i \leq p & \\ s_i \in \{0,1\} & \forall i \in L \\ 0 \leq x_{ai} \leq 1 & \forall a \in \mathcal{A}, i \in \hat{\mathcal{L}}_a \end{cases}$$

This MIP selects both a group of sensors to witness an incident and an actual sensor from the group. The fundamental structure of this formulation changes only slightly from SP, but in practice this MIP often requires significantly less memory. Specifically a grouping of k equivalent locations removes $k-1$ entries from the the objective, $k-1$ variables, and $k-1$ constraints. Every feasible solution for SP has a corresponding solution in waSP with the same sensor placement. We can always map the selected observation variable to a real sensor with the same impact. Because the impact for each incident is the same, the objective value is the same, so we can use waSP to find optimal sensor placements.

The waSP model revises SP to exploit structure in SP that can make the MIP formulation smaller. We have developed two extensions of this idea in Berry et al. [4]: witness aggregation and incident aggregation. These aggregation strategies attempt to consolidate the impact values to create smaller MIP formulations for sensor placement that approximate SP.

3.1 Witness Aggregation

We can generalize the waSP formulation to consider location values as equivalent if their impact values are approximately equal. For each incident a, consider a list of locations in $\mathcal{L}_a$ sorted by impact. A *superlocation* is a contiguous sublist of this sorted list. Generally, we group locations into a superlocation if the difference in their impact values meets a given threshold. In Berry et al. [7], we describe two ways for creating superlocations: (1) the ratio of largest to the smallest impact in the superlocation is small, and (2) the difference between the largest and the smallest impact is small. Note that the locations grouped in a superlocation for an incident are not necessarily located physically close in the network even though the contamination for incident a reaches them at approximately the same time.

Let $\tilde{\mathcal{L}}_{ai} \subseteq \mathcal{L}_a$ be the locations in the ith *superlocation* for incident a. We denote the set of superlocations for incident a by $\tilde{\mathcal{L}}_a$. Let $\tilde{d}_{ai}$ be the largest impact value for incident a if witnessed by any location in $\tilde{\mathcal{L}}_{ai}$ (that is, $\tilde{d}_{ai} = \max_{i \in \tilde{\mathcal{L}}_{ai}} d_{ai}$). And let x_{ai} be a binary variable that is 1 if incident a is witnessed by some location in $\tilde{\mathcal{L}}_{ai}$. Then the MIP for general witness aggregation is the waSP formulation where we replace d_{ai} by $\tilde{d}_{ai}$, replace $\hat{\mathcal{L}}_{ai}$ by $\tilde{\mathcal{L}}_{ai}$, and replace $\hat{\mathcal{L}}_{ai}$ by $\tilde{\mathcal{L}}_{ai}$.

We have shown that the optimal solution to a problem with ratio aggregation is guaranteed to be an approximation for the original problem with quality

proportional to the ratio. However, it is hard for a user to determine a good threshold without carefully exploring the data.

3.2 Incident Aggregation

In some cases, we can replace a pair or a group of contamination incidents with a single new incident that is equivalent. In Berry et al. [7], we describe one such strategy (called scenario aggregation in that paper for historical reasons). This aggregation strategy combines two incidents that impact the same locations in the same order, allowing for the possibility that one incident continues to impact other locations. For example, two contamination incidents should travel in the same pattern if they differ only in the nature of the contaminant, though one may decay more quickly than the other. Aggregated incidents can be combined by simply averaging the impacts that they observe and updating the corresponding incident weight α_a.

3.3 Impact

These aggregation techniques significantly improved our ability to apply MIP solvers to real-world sensor placement applications. The use of the waSP formulation is critical to solve large sensor placement problems, even on high-end workstations with large memory. For example, in Berry et al. [7] we show that aggregating witnesses with the same impacts can reduce the number of nonzeros in the MIP model by a factor of two, and it reduces the total runtime by a factor of four. Further, ratio witness aggregation and incident aggregation can be combined to formulate an approximate sensor placement formulation that reduces the number of nonzeros by a factor of 7 and the runtime by a factor of 200, while generating a solution that is within 5% of optimal.

4 The GRASP Heuristic

The MIP formulations described in the previous section cannot be solved to optimality for very large networks, even on high-end workstations with a lot of RAM. Thus, we have adapted heuristic algorithms for the p-median problem to solve SP. The current state-of-the-art heuristic for the p-median problem is the GRASP algorithm recently introduced by Resende and Werneck [20]. This GRASP heuristic is a three-phase search procedure. In the first phase, a set of high-quality solutions are generated using biased greedy construction techniques. Steepest-descent hill-climbing is then used to transform each of the resulting solutions into local optima. Finally, path relinking is used to further explore the set of solutions lying at the intersection of the resulting local optima. For a complete description of this heuristic, we refer the reader to [20].

On a series of wide-ranging tests, we observed that the GRASP heuristic was able to locate solutions to very large p-median instances (with over 10,000 facilities and 50,000 customers) in approximately ten minutes of run-time on a

modern workstation-class computer [18]. This is approximately 5-10 times faster than CPLEX when solving the waSP MIP formulation introduced in Section 3. Further, the solutions obtained by GRASP were often optimal (as verified by comparison with exact solutions to the MIP formulation). The only drawback to the GRASP heuristic involved the memory requirements, which reached 16GB of RAM for the largest instance considered. This capacity is beyond the limits of what is available in most end-user environments for which CWS design is targeted; here, the typical platform is either a 32-bit workstation (with a maximum capacity of 4GB of RAM) or a Windows workstation, which is limited to 8GB of RAM even when running on 64-bit CPUs.

The GRASP heuristic creates a dense matrix of all customer-facility "distances", as the cost of determining the decrease in "cost" during a local search move is dictated by the lookup cost of specific d_{ai} impact values. The dense matrix approach replicates information, but in doing so yields constant-time lookup of the d_{ai} coefficients. An alternative "sparse" representation simply stores, for each $a \in \mathcal{A}$, a tree containing pairs (i, d_{ai}) for all i defined for the incident a. The resulting representation yields logarithmic (in the number of defined d_{aj} for a given a) lookup costs, necessarily slowing the execution of the GRASP heuristic. However, in practice the slow-down is less than 50%, while the memory requirements are reduced by a factor of four or more.

SPOT provides variants of the GRASP heuristic using the dense and sparse storage schemes for the d_{ai}, and this optimizer has been widely used in the USEPA TEVA program. However, even with the sparse representation the largest networks considered in the USEPA TEVA program are still too large for 32-bit workstations. Other avenues have been used to reduce the problem size further for these problems, such as restricting the number of locations for sensors. These strategies may preclude the optimal solution, but they provide a practical alternative for heuristic optimization.

5 Lagrangian Heuristic

In this section, we present a Lagrangian-based bounding procedure and approximation heuristic which requires $O(n + D)$ space, where n is the number of sensor locations and D is the total number of impacts. This is an asymptotically optimal memory requirement for an in-core implementation. We use the Lagrangian-based lower-bounding method for the p-median problem described by Avella, Sassano, and Vasil'ev [1]. They give a Lagrangian model for which one can compute the optimal solution, given a set of Lagrangian multipliers, in linear space and near-linear time. Barahona and Chudak [3] give a Lagrangian formulation for the related unconstrained facility location problem, where one balances a facility opening cost with the service costs rather than limiting the number of facilities. Barahona and Chudak detail how to use subgradient search, specifically Barahona and Anbil's Volume algorithm [2], to find Lagrangian multipliers that produce progressively higher lower bounds. We adapted their method to the p-median problem. This search converges to a set of Lagrangian multipliers

for which the optimal solution to our relaxed problem is an optimal solution to the p-median LP relaxation. We then use our constrained rounding algorithm [8] to randomly select p sensor locations biased by the LP relaxation.

We now describe the Lagrangian relaxation model. As with all Lagrangian relaxation, we remove some of the constraints, leaving behind a problem that is easy to solve. We apply pressure to satisfy the constraints we have relaxed by adding penalties to the objective function. These penalties are proportional to the constraint violations. Thus there is no penalty if a constraint is met, a small penalty for a small violation, and a larger penalty for a larger violation.

We relax the first set of constraints in the SP formulation, those that require each incident is witnessed by some sensor; recall that this might be the *dummy* sensor that indicates a failure to detect the incident. This constraint is written as an equality, because that is a more efficient integer programming formulation. However, the difficult part of the constraint is insuring that at least one sensor witnesses each incident. The objective will prevent over-witnessing, so for the sake of the Lagrangian relaxation, we consider these constraints to be inequalities. For some incident a, this constraint is violated for a proposed setting of the s_i and x_{ai} variables if $\sum_{i \in \mathcal{L}_a} x_{ai} < 1$, giving a violation of $1 - \sum_{i \in \mathcal{L}_a} x_{ai}$. We weight each such violation with its own Lagrangian multiplier λ_a, which allows us to penalize some violations more than others. Adding a penalty term $\lambda_a - \lambda_a \sum_{i \in \mathcal{L}_a} x_{ai}$ to the objective for each incident a, the Lagrangian model becomes:

$$\text{(LAG) minimize} \sum_{a \in \mathcal{A}} \left(\alpha_a \sum_{i \in \hat{\mathcal{L}}_a} (d_{ai} - \lambda_a) x_{ai} \right) + \sum_{a \in \mathcal{A}} \alpha_a \lambda_a$$

$$\text{where} \begin{cases} x_{ai} \leq s_i & \forall a \in \mathcal{A}, i \in \hat{\mathcal{L}}_a \\ \sum_{i \in L} s_i \leq p & \\ s_i \in \{0,1\} & \forall i \in L \\ 0 \leq x_{ai} \leq 1 & \forall a \in \mathcal{A}, i \in \hat{\mathcal{L}}_a \end{cases}$$

For a fixed set of λ_a, we can compute the optimal value of LAG in linear space and near-linear time using a slight variation on the method described by Avella, Sassano, and Vasil'ev [1]. The optimal solution to LAG gives a valid lower bound on the value of an optimal solution to the p-median (SP) problem. This is because any feasible solution to the p-median problem is feasible for LAG. It has a zero violation for each of the lifted constraints and a value equal to the original p-median value.

Barahona and Chudak [3] describe the Volume subgradient method as applied to the unconstrained facility location problem. This method begins with $\lambda_a = 1$ for all incidents a, solves the relaxed problem, then iteratively updates the multipliers, increasing the multipliers in proportion to the violation. The updates require space and time linear in the number of variables. We modified the Vol unconstrained facility location code, available in the COIN-OR repository [9] for the p-median problem. This will converge to an optimal solution for the p-median problem.

Given a fractional solution to the p-median LP, we can treat the fractional values as probabilities and select sensors randomly according to this probability. However, one is unlikely to get precisely p sensors this way. We use the method of Berry and Phillips [8] for efficiently sampling over the "lucky" distribution where we select precisely k sensors. If necessary, we then select the dummy location.

In preliminary tests with a moderate[1] sized problem, the Lagrangian method required approximately 1/3 the space of the GRASP heuristic and usually found a solution almost as good while running up to 2.5 times longer. For example, on a problem with 3358 locations, 1621 incidents, and 5 sensors, considering four different types of objectives, the Lagrangian solver required 45Mb of memory while the GRASP heuristic required 154Mb of memory. The GRASP heuristic found the optimal solution in all four cases as verified by the MIP. The Lagrangian heuristic was within .5% of this for three out of the four objectives. Running times for GRASP ranged from 33.8 seconds to 44 seconds. The Lagrangian ran in less than 86 seconds for 3 out of 4 objectives. For the fourth objective, Lagrangian ran for 105 seconds and had a gap of 64%, showing that the Lagrangian behavior can be less stable than GRASP.

As we noted above, the Lagrangian method provides a lower bound on the value of an optimal solution to the p-median (SP) problem. Further, this lower bound is computed with less memory than an LP relaxation of SP. Thus, another practical motivation for applying the Lagrangian method is that it computes a valid lower bound on the value of solutions generated by GRASP!

We also consider the use of witness aggregation to further reduce the memory required for the Lagrangian method, particularly aggregation of locations that have the same impact values. However, we cannot embed the set-cover constraints (the second set of constraints in the waSP formulation) without altering the Lagrangian model. We can run the heuristic with the aggregated witnesses where the superlocations are not directly associated with their constituent locations. This creates a straight p-median problem for the Lagrangian solver that now no longer has the same optimal solution. Because there are fewer opportunities to witness incidents, this revised formulation has a higher optimal impact, and therefore the current Lagrangian solver does not give a valid lower bound. However, we can still compute a heuristic solution by solving this modified problem and mapping superlocations back to real locations.

We have developed a preliminary version of an aggregated Lagrangian heuristic that simply selects the first real location in a superlocation list. For a large-scale problem with 42,000 junctions, the Lagrangian heuristic required only 100Mb for the aggregation problem where we equated only witnesses of equal impact. This is a considerable reduction from the 1.8GB the Lagrangian method required with no witness aggregation, even of equal impact (the SP version). The GRASP heuristic required 17GB; there is no value for witness aggregation in the GRASP heuristic, so this is the memory requirement for the SP version. However, the objective of the Lagrangian solution is 60% worse than the solution found by GRASP. The significant reduction in space motivates more work on this

[1] This problem is the same size as those Avella et. al. call "large-scale."

aggregated Lagrangian heuristic, and we expect more sophisticated techniques for mapping from supernodes to real location will improve its performance.

6 Discussion and Conclusion

In practice, a particular sensor placement problem must be solved numerous times, e.g., to generate sensor budget versus performance trade-off curves, or to guide search toward solutions of a specific form. Consequently, the design of SPOT was initially focused on execution speed, to facilitate maximal analysis throughput. For the smaller sensor placement problems examined in the early phases of this project, emphasis on run-time achieved this goal.

However, as larger and larger problems became available, our design focus rapidly shifted from minimizing run-time toward minimizing the memory footprint for large sensor placement problems. SPOT is intended for general use by water utilities throughout the United States, most of which do not possess high-end computing platforms, and limited-memory sensor placement strategies are needed for commonly available workstations. This change in emphasis focused algorithm design and development efforts in fundamentally new, unanticipated directions.

There is a broad lesson here: a strict focus on run-time can severely limit the applicability of algorithmic techniques to real-world problems. Non-algorithmic considerations can significantly impact the practicality of an algorithmic approach. End-users consider a wide range of factors when deciding to use a computational tool, such as likely acceptance in their organization, the background of users, and required computation resources. These factors can easily outweigh algorithmic considerations like run-time efficiency. This is not a new observation, but what is surprising is that most discrete optimization research appears to be driven strictly by run-time considerations, e.g., to obtain either new best-known solutions to benchmark problems or reduce the run-time required to obtain high-quality solutions. This project illustrates that focusing on other performance factors can lead to fundamentally new algorithmic challenges, and that assessments of algorithmic strategies should consider trade-offs between factors that impact their use in practice.

Acknowledgements

We have collaborated with Regan Murray (US EPA), Rob Janke (US EPA), Jim Uber (University of Cincinnati) and Tom Taxon (Argonne National Laboratory) on the development of the TEVA-SPOT Toolkit. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94-AL85000.

Bibliography

[1] P. Avella, A. Sassano, and I. Vasil'ev, *Computational study of large-scale p-median problems*, Mathematical Programming, 109 (2007), pp. 89–114.

[2] F. Barahona and R. Anbil, *The volume algorithm: producing primal solutions with a subgradient method*, Mathematical Programming, 87 (2000), pp. 385–399.

[3] F. Barahona and F. Chudak, *Near-optimal solutions to large-scale facility location problems*, Discrete Optimization, 2 (2005), pp. 35–50.

[4] J. Berry, R. D. Carr, W. E. Hart, and C. A. Phillips, *Scalable water sensor placement via aggregation*, in Proc. Water Distribution System Symposium, 2007.

[5] J. Berry, L. Fleischer, W. E. Hart, C. A. Phillips, and J.-P. Watson, *Sensor placement in municipal water networks*, J. Water Planning and Resources Management, 131 (2005), pp. 237–243.

[6] J. Berry, W. E. Hart, C. A. Phillips, and J. Uber, *A general integer-programming-based framework for sensor placement in municipal water networks*, in Proc. World Water and Environment Resources Conference, 2004.

[7] J. Berry, W. E. Hart, C. E. Phillips, J. G. . Uber, and J.-P. Watson, *Sensor placement in municiple water networks with temporal integer prog ramming models*, J. Water Resources Planning and Management, 132, pp. 218–224.

[8] J. Berry and C. Phillips, *Randomized rounding for sensor placement problems*. In preparation. To be submitted., 2007.

[9] *Computational INfrastructure for Operations Research home page*. http://www.coin-or.org/.

[10] W. E. Hart, J. Berry, R. Murray, C. A. Phillips, L. A. Riesen, and J.-P. Watson, *SPOT: A sensor placement optimization toolkit for drinking water contaminant warning system design*, Tech. Report SAND2007-4393 C, Sandia National Laboratories, 2007.

[11] A. Kessler, A. Ostfeld, and G. Sinai, *Detecting accidental contaminations in municipal water networks*, Journal of Water Resources Planning and Management, 124 (1998), pp. 192–198.

[12] A. Kumar, M. L. Kansal, and G. Arora, *Discussion of 'detecting accidental contaminations in municipal water networks'*, Journal of Water Resources Planning and Management, 125 (1999), pp. 308–310.

[13] B. H. Lee and R. A. Deininger, *Optimal locations of monitoring stations in water distribution system*, Journal of Environmental Engineering, 118 (1992), pp. 4–16.

[14] B. H. Lee, R. A. Deininger, and R. M. Clark, *Locating monitoring stations in water distribution systems*, Journal, Am. Water Works Assoc., (1991), pp. 60–66.

[15] P. Mirchandani and R. Francis, eds., *Discrete Location Theory*, John Wiley and Sons, 1990.

[16] K. Morley, R. Janke, R. Murray, and K. Fox, *Drinking water contamination warning systems: Water utilities driving water research*, Journal AWWA, (2007), pp. 40–46.

[17] A. Ostfeld and E. Salomons, *Optimal layout of early warning detection stations for water distribution systems security*, Journal of Water Resources Planning and Management, 130 (2004), pp. 377–385.

[18] A. Ostfeld, J. G. Uber, E. Salomons, J. W. Berry, W. E. Hart, C. A. Phillips, J.-P. Watson, G. Dorini, P. Jonkergouw, Z. Kapelan, F. di Pierro, S.-T. Khu, D. Savic, D. Eliades, M. Polycarpou, S. R. Ghimire, B. D. Barkdoll, R. Gueli, J. J. Huang, E. A. McBean, W. James, A. Krause, J. Leskovec, S. Isovitsch, J. Xu, C. Guestrin, J. VanBriesen, M. Small, P. Fischbeck, A. Preis, M. Propato, O. Piller, G. B. Trachtman, Z. Y. Wu, and T. Walski, *The battle of the water sensor networks (BWSN): A design challenge for engineers and algorithms*, J Water Resource Planning and Management, (2007). (submitted).

[19] M. Propato, O. Piller, and J. Uber, *A sensor location model to detect contaminations in water distribution networks*, in Proc. World Water and Environmental Resources Congress, American Society of Civil Engineers, 2005.

[20] M. Resende and R. Werneck, *A hybrid heuristic for the p-median problem*, Journal of Heuristics, 10 (2004), pp. 59–88.

[21] L. A. Rossman, *The EPANET programmer's toolkit for analysis of water distribution systems*, in Proceedings of the Annual Water Resources Planning and Management Conference, 1999. Available at http://www.epanet.gov/ORD/NRMRL/wswrd/epanet.html.

[22] USEPA, *WaterSentinel System Architecture*, tech. report, U.S. Environmental Protection Agency, 2005.

[23] J.-P. Watson, H. J. Greenberg, and W. E. Hart, *A multiple-objective analysis of sensor placement optimization in water networks*, in Proc. World Water and Environment Resources Conference, 2004.